Vancouver Institute for Higher Learning
UBC AMS Nest Room 2312
Vancouver BC
(123) 456-7890

Computer Programming Lab

Mr. Sebastián Gil

OVERVIEW

Who doesn’t love their smartphone? More specifically, who doesn’t love the cornucopia of apps available? From games, to social media and secure banking platforms, computer programming is everywhere. In this course, we’ll explore the essential building blocks behind all computer programming platforms and demonstrate how these are implemented in the sciences and private sectors of industry. Students will learn how to read and write code in Python.

WHY PYTHON?

Traditionally, introductory computer programming courses take one of two routes: using a ‘toy language’ or an established language such as C++ and Java. The motivation for ‘toy languages’ is to not overwhelm students that are just learning to program with hard-to-read code, but this is at the expense of remaining in a ‘sandbox’ lacking any concrete real-world applications. Conversely, though C++ and Java are widely used in industry (most web platforms run on JavaScript), these may indeed prove to be too daunting for the beginner.

The popularity of Python as a programming language has skyrocketed in the past decade to become the most widely-used language in introductory programming courses in American and Canadian universities. Python code can be found in the servers of Google, NASA, the CIA, Dropbox and Netflix!

By being clearly legible while retaining flexibility and utility, Python is an excellent entry into the world or programming. Best of all, it is completely free! This means that anybody can access the plethora of open source modules and implementations of the language.

COURSE OBJECTIVES

By the end of the course you should be able to

1. Appreciate the prevalence of computer programming across all areas of contemporary society.
2. Interpret code written in Python and be able to provide concise explanations for what simple programs do, which includes reading other people's code.
3. Distinguish between the different types of data structures used in programming environments and write iterative procedures using if statements, for loops and while loops.
4. Access documentation for how to use built-in functions to improve your own code through the standard libraries and external resources such as StackExchange.
5. Explain how the roles of front end and back end software engineeris complement each other and how their work makes programs user-friendly.

COURSE OUTLINE

Each unit is expected to cover a month of the term.

Unit 1 -- The Vocabulary of Computer Programming

This unit begins by introducing students to the world of computer science and the diverse applications that it has across all commercial and academic sectors of society. Students will learn how to install an IDLE (you'll learn what this means!) and write their first programs. The building blocks of any programming language, such as 'floats' and 'strings' will be introduced as well.

Unit 2 -- Control Structures, Functions and Modules

Having gained confidence in how to navigate within an integrated developing environment, this unit will focus on creating variables to store information and how these variables can be manipulated to perform a variety of tasks. An in-depth overview of the essentials of programming-- if statements, for and while loops--will be done as well. Finally, the unit will conclude in an explanation of how to reuse code by writing functions and learning to access pre-built functions from the modules of standard libraries.

Unit 3 -- Algorithms and Data Science: The Back End Engineer

So far, students have learned the bare-bones structure behind all computer languages. The remainder of the course will focus on how software engineers in the real world accomplish tasks. To this end, the front/back end distinction is used, which can be illustrated by analogy to a sports car. If a computer program is a fancy car, the backend engineer is the person responsible for making sure that the engine runs, that the oil is replaced and knows how to fix the car if it gets broken. Backend engineers write algorithms to optimize computation time and access databases as efficiently as possible. They run servers and administrate user logins and other security protocols for companies.

Unit 4 -- Web Development and Graphic Design: The Front End Engineer

Back to our sports car analogy, the front end engineer ensures the car remains clean, flashy and beautiful. This involves taking care of the chassis of the car, adding new coats of paint and advertising similar cars to others. Front end engineers focus on developing websites and apps while ensuring that they remain beautiful and easy to use. End user accessibility is the goal of front end engineers: they know their job is done when people can easily drive the car without requiring any knowledge of ‘what’s under the hood.’

GRADE DISTRIBUTION

Participation -- 25%

Coding is by no means a passive task! Computer programming is a unique cognitive task in the sense that it involves critical thinking, analysis and ongoing debugging. As such, one cannot expect to learn how to program just by passively watching others do it, or merely by reading code. Students are expected to follow along with the instructor and to experiment with all the new commands that they are learning.

Midterm Exam -- 25%

Halfway throughout the course there will be a short exam consisting mainly of multiple-choice questions and fill-in the blank type of problems. The objective of the exam is to reinforce the students’ newly acquired knowledge of how to write simple functions and check for any syntax errors. Students that practice regularly the concepts learned in the course should have no difficulty passing the exam!

Term Project -- 50%

In the remaining two months following the midterm, the course will focus on an overview of front end and back end software engineers and their role in industry. In order to fully synthesize all the skills acquired throughout the course, students will develop and implement their own project on either of these disciplines, with assistance from the instructor as necessary. Students choosing the front end option can develop (for example) a simple website in the 2.0 web framework, while students choosing the back end option can develop a simple security protocol for password encryption.